MODULE DESCRIPTION

Module code	**ID1WI1**
Module name	**Wstęp do informatyki**
Module name in English	**Introduction to Computer Science**
Valid from academic year	**2012/1013**

MODULE PLACEMENT IN THE SYLLABUS

Subject	**Computer Science**
Level of education	**1st degree** *(1st degree / 2nd degree)*
Studies profile	**General** *(general / practical)*
Form and method of conducting classes	**Full-time** *(full-time / part-time)*
Specialisation	
Unit conducting the module	**The Department of Electronics and Intelligent Systems**
Module co-ordinator	**Filip Rudziński, PhD, Eng.**
Approved by:	

MODULE OVERVIEW

Type of subject/group of subjects	**Major** *(basic / major / specialist subject / conjoint / other HES)*
Module status	**Compulsory** *(compulsory / non-compulsory)*
Language of conducting classes	**Polish**
Module placement in the syllabus - semester	**1st semester**
Subject realisation in the academic year	**Winter semester** *(winter / summer)*
Initial requirements	**No requirements** *(module codes / module names)*
Examination	**Yes** *(yes / no)*
Number of ECTS credit points	**5**

Method of conducting classes	Lecture	Classes	Laboratory	Project	Other
Per semester	**30**		**15**		

TEACHING RESULTS AND THE METHODS OF ASSESSING TEACHING RESULTS

Module target	The aim of the module is to teach students the ability of computer program implementation with the use of C and C++ programming (together with the paradigms of imperative, procedural, and object-oriented programming).

Effect symbol	Teaching results	Teaching methods *(l/c/l/p/other)*	Reference to subject effects	Reference to effects of a field of study
W_01	A student has basic knowledge as regards fundamental notions concerning computer science, including numeral systems as well as information processing techniques (using basic and complex data types).	l	K_W06	T1A_W04 T1A_W07
W_02	A student understands the notion of an algorithm and is familiar with the issue concerning algorithm design as well as the analysis of their computational complexity; a student is also familiar with typical algorithmic tasks and the strategies of solving them.	l	K_W07	T1A_W04 T1A_W07
W_03	A student is aware of important selected computer science issues covering, among other things, paradigms and programming languages, cryptography, artificial intelligence, and concurrent programming.	l	K_W11 K_W13	T1A_W04 T1A_W07
U_01	A student has mastered compiler handling, including the support system and the documentation of Pascal as well C++ programming languages.	l	K_U01	T1A_U01
U_02	A student has mastered arithmetic and bit manipulations on data with the use of NKB and U2 numeral systems. A student is able to design a complex structure of data type and analyse its binary representation in computer memory.	l	K_U12	T1A_U09 T1A_U16
U_03	A student is able to design algorithms with the graphical method and analyse their computational complexity. A student can also implement the designed algorithms with the use of the selected programming language.	l	K_U13	T1A_U09 T1A_U16
K_01	A student is aware of the impact concerning the selected tools from the field of computer science on the development of other branches of science.	l	K_K02	T1A_K02

Teaching contents:

Teaching contents as regards lectures

Lecture number	Teaching contents	Reference to teaching results for a module
1	**Basic notions** Information and information units of measurement. The elements of logic and Boolean algebra.	W_01
2	**Numeral systems** Binary representations of integers (natural binary code and augmented	W_01

	systems) as well as real numbers (fixed- and floating-point representations). Conversions on integers between systems with the following bases: 2, 8, 10, and 16.	
3	**Basic data types** The kind of types representing integers and real numbers in the selected programming languages. Logic data types. ASCII and UNICODE sign standards.	W_01
4	**Elementary operation using basic data types** Arithmetic, logic, and bit manipulations using restricted byte length integers. Fixed-point and floating-point arithmetic.	W_01
5	**Complex data types** One- and two-dimensional arrays of integers/real numbers (vectors and matrices). Addressing array elements. Sign chains. Structures (records) and unions. Abstract data types (an outline): lists, trees, and graphs.	W_01
6	**The analysis of binary representations of complex data types** The structure of a two-dimensional array using vectors and an address array. The mechanism of aligning box dimensions in records. Conversions and data types casting. Special representations (colours, complex numbers, unlimited byte length integers, date and time representations).	W_01
7	**Graphical methods of designing algorithms** The definition of an algorithm and the notion of control flow. Graphical symbols utilised in flow diagrams. Control instructions and their graphical representations. Input/output operations in data validation.	W_02
8	**The diagrams of basic algorithmic structures** The algorithms of enumerative browsing/searching processing data contained in one- or two-dimensional arrays (summing element values, determining an element with maximum/minimal value, etc.). Basic sorting algorithms. Recurrence algorithms.	W_02
9	**Computational complexity of algorithms** The notion of time and memory complexity. Big O notation. Little-o notation. Turing machine. Deterministic and nondeterministic algorithms.	W_02
10	**The classes of algorithmic tasks** P-class and NP-class assignments. Algorithmic strategies. The examples of typical combinatorial tasks.	W_02
11	**Programming languages and paradigms** Basic programming paradigms (imperative, procedural, object-oriented, functional, declarative, event-driven, and logic). Discussing the selected programming languages, i.e. assemblers, Pascal, , Pascal, C/C++, Java, C#, Haskell, Ada, Lisp, and Prolog.	W_03
12	**The elements of cryptography** Symmetric and asymmetric cryptographic algorithms, hash functions, Pseudo-Random Number Generators (PRNG).	W_03 K_01
13	**Introduction to concurrent programming** The notion of a thread and process. Fine-grained and coarse-grained computations. Computational clusters and cloud computing. UMA and NUMA memory architectures. Thread and process synchronisation. Inter-process	W_03

	communication.	
14	**Modern developmental directions of computer science: artificial intelligence, part 1** An outline of the selected tools as regards computational intelligence (artificial neural networks and genetic algorithms) as well as their applications.	W_03 K_01
15	**Modern developmental directions of computer science: artificial intelligence, part 2** An outline of the selected tools as regards computational intelligence (rule-based fuzzy systems) as well as their applications.	W_03 K_01

Teaching contents as regards laboratory classes

Laboratory class number	Teaching contents	Reference to teaching results for a module
1	**An introduction to laboratory classes** Familiarising students with the programming environment (Embracadero Delphi, C++ Builder or Microsoft Visual Studio) together with the documentation concerning laboratory classes.	U_01
2	**Arithmetic operations in the selected numeral systems** Arithmetic operations using NKB and U2 positional numeral systems (numbers with a byte length limited to 8, 16, and 32 bytes) as well as real numbers, including the analysis of summing and subtracting numbers taking the acceptable value range into consideration.	U_02
3	**Bit manipulations in the selected numeral systems** Bit manipulations using NKB and U2 positional numeral systems (numbers with a byte length limited to 8, 16, and 32 bytes), including multiplying and dividing numbers by multiple values of 2, examining the sign and number parity by setting, resetting, negation, and examining single values or bit groups in integers.	U_02
4	**Examining binary representations of complex data types** Examining a binary representation of one- and two-dimensional arrays of integers, data structure (record), and data unions.	U_02
5	**Algorithm designing with the graphical method** Designing the selected enumerated algorithms with the graphical method: determining an arithmetic/geometric mean, maximum/minimum value, the number of elements meeting the requirements set (from among the elements of one- or two-dimensional arrays).	U_03
6	**Examining computational complexity of algorithms** Determining the characteristics of time computational complexity of the selected sorting and data searching algorithms (included in complex data structures).	U_03
7	**Algorithm implementation using the PASCAL language** The implementation of algorithms designed with the graphical method (as part of laboratory classes No 5) using either the Pascal or C++ language.	U_03

The methods of assessing teaching results

Effect symbol	**Methods of assessing teaching results** *(assessment method, including skills – reference to a particular project, laboratory assignments, etc.)*
W_01 W_02 W_03 K_01	A written examination on completing the classes.
U_01 U_02 U_03	Written tests (5) at the beginning of laboratory classes No 2-6.

STUDENT'S INPUT

ECTS credit points		
	Type of student's activity	**Student's workload**
1	Participation in lectures	**30**
2	Participation in classes	
3	Participation in laboratories	**15**
4	Participation in tutorials (2-3 times per semester)	**1.5**
5	Participation in project classes	
6	Project tutorials	
7	Participation in an examination	**1.5**
8		
9	**Number of hours requiring a lecturer's assistance**	**48** *(sum)*
10	**Number of ECTS credit points which are allocated for assisted work** *(1 ECTS credit point=25-30 hours)*	**1.92**
11	Unassisted study of lecture subjects	**0.96 (24 hours)**
12	Unassisted preparation for classes	
13	Unassisted preparation for tests	
14	Unassisted preparation for laboratories	**0.96 (24 hours)**
15	Preparing reports	
16	Preparing for a final laboratory test	
17	Preparing a project or documentation	
18	Preparing for an examination	**1.16 (29 hours)**
19	Preparing questionnaires	
20	**Number of hours of a student's unassisted work**	**77** *(sum)*
21	**Number of ECTS credit points which a student receives for unassisted work** *(1 ECTS credit point=25-30 hours)*	**3.08**
22	**Total number of hours of a student's work**	**125**
23	**ECTS credit points per module** *1 ECTS credit point=25-30 hours*	**5**
24	**Work input connected with practical classes** *Total number of hours connected with practical classes*	**40.5**
25	**Number of ECTS credit points which a student receives for practical classes** *(1 ECTS credit point=25-30 hours)*	**1.62**